Server Network I/O Acceleration

Fundamental to the Data Center of the Future

Packets are now the predominant vehicle used to transport data through a network. Today's system architectures, however, were not designed for the packet volumes, bandwidth speeds and intelligent network services that will be found in tomorrow's data center. The poor packet processing efficiency of most data center equipment poses a significant challenge to the data center of the future. This white paper provides an overview of the problem and introduces Intel Research and Development's (R&D) efforts into server network I/O acceleration.

A primary tenet of Intel R&D's research into server network I/O acceleration is that efficient, scalable, packet processing can be achieved in a server without the addition of discrete silicon to assist or offload the CPU. Inefficient server network I/O is not a weakness found in a single element or subsystem within a server. Therefore, a solution will require a system-wide approach – changing the microprocessor, chipset and system software – to achieve a scalable solution.

Contents

Introduction

On the heels of each leap in network bandwidth, the combination of advancement in CPU frequency, cache architecture, memory device speed, and bus transfer rates achieved robust application and packet handling performance. Yet, researchers have long been aware that packet volumes and bandwidth would someday require packet processing performance improvements or systems would bog down under the load. The historical gains in system technologies have masked an inefficient process to create and terminate packets. Multiple interrupts, inefficient protocol stack software, multiple system memory accesses and significant operating system resource scheduling consume tremendous numbers of CPU cycles per packet. Until now, system technology gains forestalled any negative impact on datacenter performance allowing enterprise IT organizations to focus their resources elsewhere.

But, the packet processing bottleneck continues to narrow. Packet volumes are growing faster than ever and Ethernet bandwidth marches on with 10 Gbps services being deployed in a growing number of locations. The use of TCP/IP packet data, rare in the 80s and early 90s, skyrocketed with the commercialization and growth of the Internet. Today, nearly all data moves between systems in the form of a packet. Soon, the increasing use of Web services, the adoption of IP addresses for consumer goods, sensor networks and RFID tags, plus greater use of IP storage will generate an overwhelming challenge for packet processing in the data center. Meanwhile, the historical pace of technology providing ample server network I/O is beginning to sputter. Without system-wide changes packet volume and bandwidth are well on the way to overwhelming the 20 year old network stack we rely on to send and receive packets. Intel R&D is defining new technologies to improve on-CPU, system-based packet processing efficiency and increase overall application throughput. Each technology can be implemented independently to reduce system overhead but working together they deliver exceptional performance. Broad commercial deployment of servers using these technologies to handle multi-gigabit network connections is anticipated in the second half of this decade.

The Problem

The result of higher packet volumes and faster rates of arrival is a direct increase in CPU overhead for networking and a reduction in available cycles for applications. Server application throughput is throttled and datacenter performance degrades.

Studies of packet processing workloads indicate CPU cycles are consumed in three broad categories (see Figure 1):

• **System Overhead** — Cycles consumed in buffer management, operating system kernel ring transitions, scheduling and interrupt handling routines occur with each packet. These system management and oversight activities are themselves software threads that consume processor cycles and must swap in and out of the processor hardware context with the user applications, protocol stacks and other system functions.

• **Memory Access** — System memory accesses are the longest latency tasks within a system outside of disk drive access and stall the CPU until the read or write request is complete. Packet data, header, descriptor and transmission control parameters are held in system memory requiring five system memory accesses to process a single packet. Each is read and placed in CPU cache to process the packet and concludes with packet data written to the application buffer.

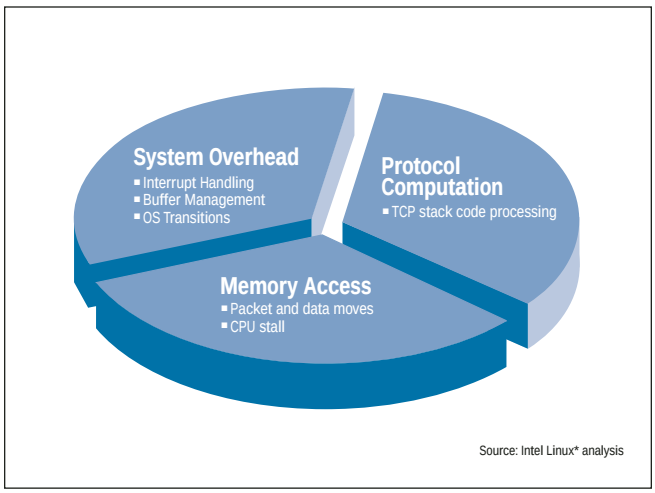

Figure 1: Server Network I/O Acceleration Bottlenecks

• **Protocol Computation** — Each packet sent or received consumes CPU cycles to process the TCP/IP stack code. Commercially available protocol stacks are based on algorithms and code developed in the early years of networking with new functions added over time. Inefficiencies have built up over time that extend the processing time for the vast majority of packets.

Today, each category consumes approximately one third of the total cycles per packet. Cycles consumed in system overhead and protocol computation have remained relatively flat since the inception of Ethernet. Increasing processor frequency helps these categories and any reduction in CPU cycles delivers a direct benefit. On the other hand, cycles consumed by system memory reads and writes are on the rise and increasing processor frequencies exacerbate the problem. To achieve a significant boost in packet handling performance no single category or symptom can be targeted.

There's been a general rule of thumb that it takes, at best, 1 MHz of CPU processing to handle 1 Mb of network bandwidth. This was a reliable tool just as long as CPU and memory performance gains kept pace with bandwidth. But memory bandwidth and access latency have fallen off their historical rate of improvement forming a growing gap with CPU performance. (See Figure 2). One effect of this emerging gap is that processor frequency gains translate into increasingly more cycles spent waiting on memory. Alongside the memory gap each ten-fold increase in network bandwidth requires a CPU to act on each arriving packet in one-tenth the time just to keep pace with packet arrival. At 10 Gbps, new packets arrive faster than a system can process a packet. This increases the likelihood of dropped packets and defeats the value of providing greater bandwidth to a server.

Seeking Solutions

The process of processing packets must be reconsidered. Emerging industry solutions such as RDMA and TCP offload engines (TOE) shift protocol stack processing from the host CPU. But these approaches do not solve the whole problem, simply a piece of it. TOE, for example, tackles the protocol processing overhead problem by offloading it from the CPU and running it on specialized silicon on or near the NIC. TCP packet order and acknowledgement by the TOE relieves the CPU of protocol instructions and a share of interrupt volume. But this addresses only a segment of the problem. Even if the protocol processing is enhanced, tremendous overhead remains due to multiple data accesses to system memory, interrupt impact and resource scheduling overhead.

Server network I/O acceleration relies on many components within a system. Any proposed solution must consider how each element used in the process, from the NIC to the CPU and chipset to the protocol stack and operating system, can be improved. Intel's server network I/O acceleration research

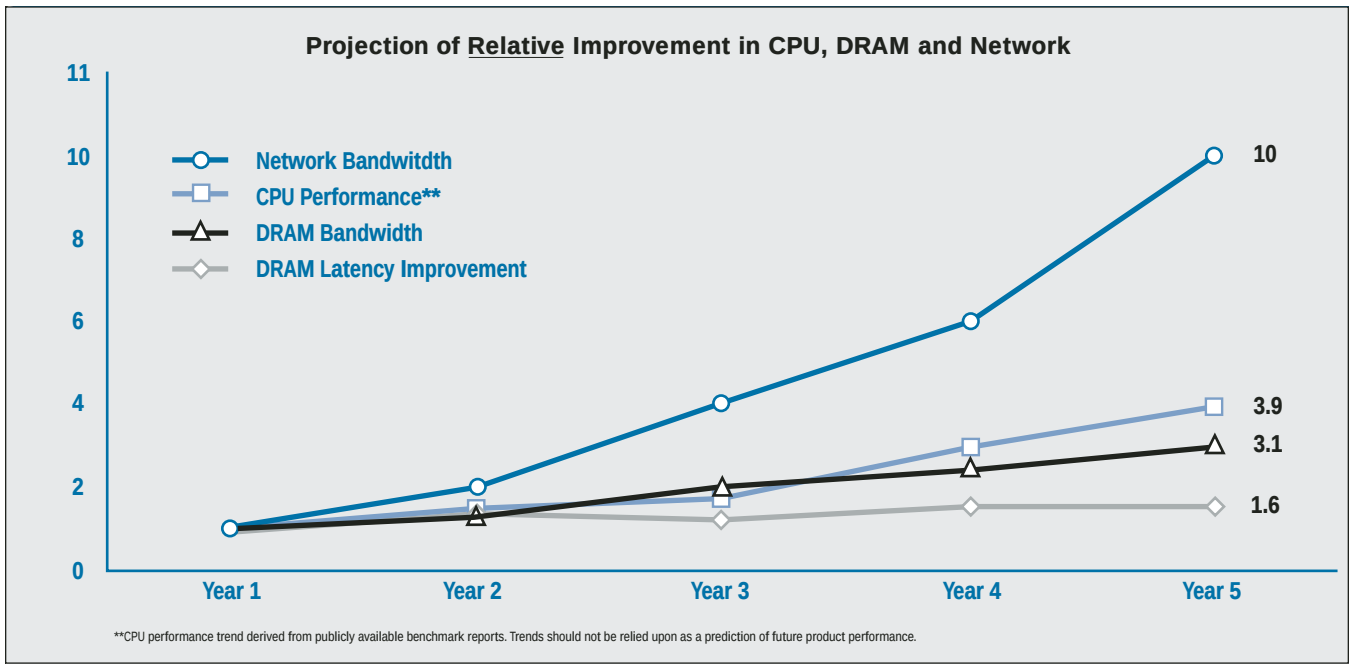

Figure 2: Rate of Technology Improvement

views these building blocks as interconnected elements that require a system-level solution in order to provide peak application throughput performance.

Server Network I/O Acceleration Technologies

Server performance is defined by application throughput the number of transactions, computations, etc. a server provides for an application. Peak application throughput, in turn, depends on data in local caches for continuous CPU activity – the same data that moves to and from the system as packets. But packets require CPU cycles and memory transactions that interfere with the very application they intend to support. The focus of Intel's server network I/O acceleration research is to improve application throughput on server systems by more efficiently delivering data to and from the application via a network connection.

Server network I/O acceleration requires an efficient path between the network interface, application memory and CPU cache. The research challenge is to utilize network bandwidth growth for a corresponding gain in user application throughput. This will require flexible system architectures to support today's range of IP-based protocols as well as those in development for next generation services. In addition, the solution must scale with growing enterprise computational demand and greater interdependence of datacenter resources.

The exploration of various server network I/O acceleration technologies by Intel R&D takes into account that TCP/IP is a data flow application. This means that there is no "natural" way to use cache memory to exploit spatial or temporal localities that reduce memory access latencies.

For example, a server platform trying to run at line rates has to deal with packet arrival rates on a high-speed (10 Gbps) network that are effectively smaller than the sum of main memory access and compute time latencies. Further complicating matters, much of the CPU time spent on packet processing is not productive. Long latency memory access forces the CPU to stall forward progress while waiting on a memory request to complete.

However, the very nature of network flow allows multiple packets to be operated on simultaneously. The approaches being explored by Intel R&D exploit this parallelism to achieve the networking performance of tomorrow's enterprise. Rather than focusing only on the CPU, chipset, NIC, or operating system, Intel is looking at the complete packet process bottleneck and creating solutions that work at the system level.

Five primary research vectors are yielding positive results in reducing the hurdles to server network I/O acceleration:

- Network Stack Affinity/Partitioning
- Protocol Stack Optimization
- Direct Cache Access
- Lightweight Threading
- Asynchronous Low-cost Copy

Modeling, simulation and prototyping indicates each delivers an improvement in one or more of the broad categories: system overhead, memory access or protocol computation. In combination, these technologies result in an order of magnitude reduction in CPU cycles consumed in packet processing. As stand-alone technologies, developers may implement the technologies as they become available to meet individual product development schedules and market requirements.

Network Stack Affinity/Partitioning

Future generations of Intel® processors will offer multiple processor cores as well as threads providing an opportunity to use these additional processing engines in new ways. One novel approach that delivers server network I/O acceleration is to partition an application from the operating system workloads that intrude on the application. "OS Intrusion" is a measure of the impact that an operating system has upon the performance of an application.[1] For server network I/O acceleration, the idea is to reduce the OS intrusion associated with network I/O operations by segregating that workload on separate processing engines. A simple queue based interface between the I/O task and the application minimizes CPU overhead and requires no change to the application. Experiments with this approach indicate far greater efficiency than today's duplication of the network stack across multiple CPUs in a system or cores in a single CPU.

Going beyond the partitioning of these workloads, research is underway at Intel to investigate processor and software architecture enhancements to take even greater advantage of multiple CPU cores. Core partitioning, permanent or dynamic, based on system workload, would allow load balancing across processor cores. The affinity of distinct system tasks with a core, or subset of cores can reduce operating system intrusion on the user application(s).

1. Piglet: A Low-Intrusion Operating System Architecture, Steve Muir, Distributed Systems Lab, University of Pennsylvania; www.acu.rl.ac.uk/msn99/Talks/Steve_Muir.pdf

Protocol Stack Optimization

The TCP/IP stack remains fundamentally unchanged since it was introduced in the early 1980s. As CPU and system architectures evolved over the last twenty years, the TCP/IP stack has been slow to take advantage of these features. When new features were utilized or protocol extensions were defined, the implementation was often "glued" into the existing stack software. Consequently, over time the venerable TCP/IP stack became inefficient and cumbersome when compared to today's hardware capabilities for fast protocol execution.

The Intel R&D team sought an optimal path through the protocol stack for a majority of packets. System and processor architecture features were considered and applied. Separate data and control paths were implemented to optimize processing of the header from the packet payload. Cache-aware data structures were used increasing cache hits. Exception testing was streamlined to reduce path length. Finally, modern coding and compiler techniques were applied. The result is an optimized stack that reduces protocol processing cycles by nearly fifty percent.

Direct Cache Access

Another avenue being explored by Intel research seeks to reduce the system memory reads required for each packet. Direct Cache Access research examines the opportunity to place the packet header and descriptor directly into a processor's cache for immediate access by the network stack process. Packets arriving at the NIC would have the header and the descriptor written into the lowest level cache. Packet payload would be written to the cache or to the NIC buffer or both to take advantage of other emerging data movement technologies. With Direct Cache Access the objective is to take full advantage of high-speed cache and eliminate processor cycles required to read packet headers and descriptors from system memory. With Direct Cache Access the number of memory accesses to process a packet drops to three from five, a forty percent reduction in memory access processing cycles.

Lightweight Threading

As noted earlier, each packet uses five system memory accesses which stall the processor during each transaction: reads to load header, descriptor and TCB data into cache and then a read and write to copy the data from NIC buffer to the application buffer. Each memory stalls waste thousands of CPU cycles that might be used for system processes or other applications.

Lightweight threading is an application of helper threads for network stack processing to mitigate the impact of a system memory event. Rather than initiating the network stack for each packet the network stack is revised as a multithreaded application and each new packet initiates a new thread in the stack. Lightweight threading enables a faster switch to an alternate software thread – another packet – upon a system memory access event. Rather than providing multiple hardware contexts in a processor like Hyper-Threading (HT) Technology[2] from Intel, a single hardware context contains the network stack with multiple software-controlled threads. When a packet thread triggers a memory event a scheduler within the network stack selects an alternate packet thread and loads the CPU execution pipeline. Processing continues in the shadow of a memory access. The most recent packet thread continues to run to completion or until another memory event occurs. The benefit of lightweight threading is that CPU cycles are used for valuable code execution, improving server application throughput. Stall conditions, triggered by requests to slow memory devices, are nearly eliminated. While multiple memory moves still occur for each packet, their impact on performance is dramatically reduced.

Asynchronous Low-cost Copy

The processing of a packet concludes with a copy of the payload data from the NIC buffer in system memory to an application buffer also located in system memory. Here again the CPU is consumed until the completion of the data copy via a memory read into CPU cache followed by a memory write. To return these cycles to productive application workloads Intel R&D is developing Asynchronous Low-cost Copy technology or ALCC.

ALCC requires the addition of a small amount of logic to the platform to simplify a system memory copy for the CPU. The breakthrough technology within ALCC is the ability of the CPU and ALCC logic to initiate and close a memory copy at "low-cost" or with far fewer cycles than the CPU would typically consume to conduct the copy itself. Then, when the CPU initiates a system memory copy the request is trapped by the ALCC logic. The ALCC will carry out the memory read and write independent of the CPU and upon completion give notice to the CPU to close out the copy request. During an ALCC managed copy, the CPU can process an alternative thread such as another packet returning precious cycles to productive work.

Conclusion

In order to make tomorrow's data centers more cost-effective and flexible, Intel researchers are studying the fundamental challenges to efficient packet processing in servers.

At the top of the list is the widening gap between line speeds and memory bandwidth that impacts CPU performance. Handling network traffic at multi-gigabit rates requires significant computational power and platform architectures designed with network processing in mind.

Intel is taking advantage of the fact that network flows lend themselves naturally to concurrency of processing, a parallelism that can be exploited in hardware. Intel researchers are investigating five interacting technologies to improve server network I/O acceleration for multi-gigabit network I/O performance. They include:

- Network Stack Affinity/Partitioning
- Protocol Stack Optimization
- Direct Cache Access
- Lightweight Threading
- Asynchronous Low-cost Copy

The combination of these technologies in future server platforms are expected to yield an order of magnitude gain in packet processing throughput using one tenth the CPU cycles consumed in today's architectures. Since these technologies are not tightly coupled developers are afforded the opportunity to include the right mix to meet product and market requirements and implement as product schedules permit.

The answers to these and other critical questions surrounding packet processing will allow Intel to continue providing advanced building blocks that leading system solution vendors will use to enable the efficient, robust and secure data center of the future.

For more information on this topic and other Intel research in the area of performance networks please go to: www.intel.com/technology/perfnet

2. Hyper-Threading Technology requires a computer system with an Intel® Pentium® 4 processor supporting Hyper-Threading Technology and an HT Technology enabled chipset, BIOS and operating system. Performance will vary depending on the specific hardware and software you use. See http://www.intel/info/hyperthreading/ for more information including details on which processors support HT Technology.

Copyright © 2004 Intel Corporation. Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Printed in USA/0504/HBD/MG/PDF